4-2009

Localization Using Extended Kalman Filters in Wireless Sensor Networks

Ali Shareef

Yifeng Zhu

13

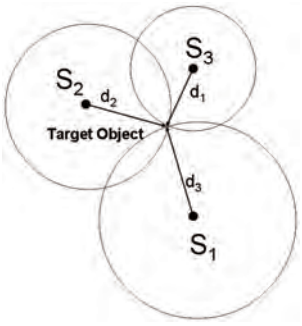
Localization Using Extended Kalman Filters in Wireless Sensor Networks

Ali Shareef and Yifeng Zhu
Electrical and Computer Engineering
University of Maine
United States of America

1. Introduction

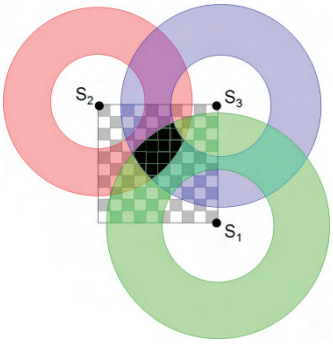
Localization arises repeatedly in many location-aware applications such as navigation, autonomous robotic movement, and asset tracking. Analytical localization methods include *triangulation* and *trilateration*. Triangulation uses angles, distances, and trigonometric relationships to locate an object. Trilateration, on the other hand, uses only distance measurements to identify the position of the target. Figure 1A, describes a simple example of trilateration. Using three reference points S_1, S_2, and S_3 with known locations and distances d_1, d_2, and d_3 to the target object, the object can be located at the intersecting point of the three circles.

However, in a dynamic system where distance measurements are noisy and fluctuate, the task of localizing becomes difficult. This can be seen in Figure 1B, where with fluctuating distances, regions within the circles become possible locations for the tracked object. In this case, rather than the object being located at a single point at the intersection of the circles as in Figure 1A, the object can be located anywhere in the dark overlapped region in Figure 1B.



A.

Fig. 1A. Trilateration



B.

Fig. 1B. Trilateration with noise

This uncertainty due to measurement noises renders analytical methods almost useless. Localization methods capable of accounting for and filtering out the measurement noises are

Source: Kalman Filter: Recent Advances and Applications, Book edited by: Victor M. Moreno and Alberto Pigazo,
ISBN 978-953-307-000-1, pp. 584, April 2009, I-Tech, Vienna, Austria

desired. The method by which the distance measurements are carried out determines the sources of noise in these measurements.
Typically devices known as "beacons" are placed at known locations and emit either radio or acoustic signals or both. It is possible for a "mobile node" to determine the distance to a beacon by using properties of these signals such as the signal strength of the RF signal, Received Signal Strength (RSS) (Patwari, et al., 2003). Other methods utilize both RF and acoustic signals by computing the time difference between the reception of an RF pulse and an acoustic pulse generated by a beacon. RF signals travel at the speed of light and the time it takes for a RF signal to get to a mobile node is almost instantaneous and can be considered zero while the propagation time of an acoustic signal in air is much longer (about 1.13 ft/ms at room temperature). This is exactly the way the Crickets (Priyantha, et al., 2000) operate where the RF and ultrasound signals are emitted simultaneously by the beacons. The mobile node computes the distance by using the time it takes for the first instance of the acoustic pulse to reach the sensor after the RF signal. However, wave reflection is common to both RF and acoustic signals and it is possible that the mobile node erroneously identifies a pulse due to the reflected wave of the original pulse as a new pulse (Ward, et al., 1997). This, of course, results in skewed distance measurements.

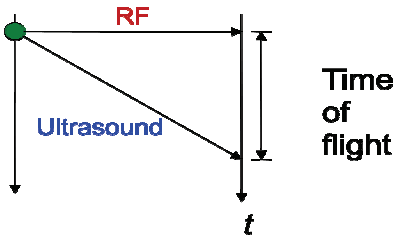


Fig. 2. Distance computation using time of flight between RF and Ultrasound signals

The Kalman filter is an iterative state estimator and it is widely used to locate an object with noisy measurements. However, the formalization of the filter states will dictate its computational and memory efficiency. The computation and memory efficiency of the Kalman filter is of concern because it is frequently implemented on embedded systems with limited computation and memory resources. A formalization of the Kalman filter with excessive states will potentially overburden a sensor system and result in a performance decrease. A trade-off must be made that balances the timely computation of the position and the desired accuracy necessary for the application. This chapter will examine the accuracy, robustness, computational and memory efficiencies of localization utilizing three different formalizations of the Kalman filters, including the position (P), position-velocity (PV), and position-velocity-acceleration (PVA) models. Our experiments conclude that all Kalman filter models are robust and are capable of handling large erroneous data. While the P model has the least computation and memory complexity, one of these models may have the best localization accuracy than the others depending upon the motion of the moving object.

2. Related work

The Active Badge Location System (Want, et al., 1992) is often credited as one of the earliest implementations of an indoor sensor network used to localize a mobile node. While this

system, utilizing infrared signals, was only capable of localizing the room that the mobile node was located in, many other systems based on this concept have been proposed.
The Bat system (Harter, et al., 1999), much like the Active-Badge System, also utilizes a network of sensors. A central controller broadcasts an RF query to a mobile node and resets a network of serially linked receivers at the same time. The mobile node responds by emitting an ultrasonic pulse which is picked up by the receivers. The time it takes for the ultrasound pulse to reach different receivers in the network indicates the distance the mobile node is from those receivers and the position of the mobile node can then be trilaterated.
Researchers at MIT have utilized similar concepts from the Bat System in their Cricket sensors, albeit using a more decentralized structure. However, one drawback to the Crickets is the risk of collisions during the RF and Ultrasound transmissions between different beacons. The Cricket Location System (Smith, et al., 2004) uses a hybrid approach involving the use of an Extended Kalman filter, Least Square Minimization to reset the Kalman filter during the Active state, and Outlier Rejection to eliminate bad distance readings.
Other researchers at MIT have proposed another method of localization utilizing the Cricket system exploiting properties of robust quadrilaterals to localize an ad-hoc collection of sensors without the use of beacons (Moore, et al., 2004).
It is also possible to localize optically as in the HiBall head tracking system (Welch, et al., 2001). Arrays of LEDs flash synchronously, and cameras capture the position of these LEDs. The system utilizes information about the geometry of the system and computes the position.
Localization using signal strength of RF signals has been studied extensively, (Alippi, et al., 2005) and (Patwari, et al., 2003) are all examples of methods that were devised using this approach.
Neural networks have also been used for localization (Shareef, et al., 2008). In fact, neural networks have been shown to perform better than the Kalman filter in a low-noise environment. However, neural networks suffer from weak self-adaptivity and have a high over-head due to training costs. A neural network, once trained for a particular set of parameters can only be used in the scenario corresponding to those parameters. If the number or the position of beacons or the size of the localization area was to change, then the neural network would have to be trained again.

3. Kalman filter applied to localization

The Kalman filter is an iterative approach that uses prior knowledge of noise characteristics to account for and filter out the noise. However, problems arise when attempting to model noise. Attempts at measuring noise are only approximations and do not indicate the real distribution of the noise. The Kalman filter can only be used for linear stochastic processes and for non-linear processes the *Extended Kalman Filter* (EKF) must be used. The assumption with these two methods is that the process and noise measurements are independent, white, and with normal probability.
There are different parameters that the EKF can use in modeling the trajectory of a moving object. It is possible to model the motion of an object using just the state of the X and Y position to obtain the position (P) model of the Kalman filter. The velocity can also be incorporated in the state in addition to the position to form the position-velocity (PV) model.

Of course, if acceleration is included also, this results in the position-velocity-acceleration (*PVA*) model.
In two-dimensional space, the distances to three known beacons returned by a sensor can be related to the position of this sensor (x, y) using the distance formula given in (1),

$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2} \tag{1}$$

where (x_i, y_i) is the coordinations of beacon *i* (*i* = 1, 2, and 3).
A way of modeling motion is by setting up a linear system composed of the kinematics equations for each dimension of tracked motion. The following example of a linear system describes an object's two-dimensional motion using the position, velocity, and acceleration (PVA) at time step *k*.

$$\begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \\ \ddot{x}_k \\ \ddot{y}_k \end{bmatrix} = A \cdot \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \\ \ddot{x}_{k-1} \\ \ddot{y}_{k-1} \end{bmatrix} + B \cdot \begin{bmatrix} u_{xk} \\ u_{yk} \end{bmatrix} + Q \tag{2}$$

Equation (2) can be written in a simpler way.

$$X_k = A \cdot X_{k-1} + B \cdot U_k + Q$$

The state transition matrix A arises from the respective kinematics equations. For a *PVA* model, the A matrix becomes:

$$A = \begin{bmatrix} 1 & 0 & T & 0 & \frac{1}{2}T^2 & 0 \\ 0 & 1 & 0 & T & 0 & \frac{1}{2}T^2 \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

The u_{xk} and u_{yk} are the inputs to the system, and the B matrix is the input matrix. However, the input kinematics parameters of the moving object to be tracked are not known so the u_{xk}, u_{yk}, and B can be dropped from the linear system. If this information had been known, then there would be no need to, "track" the object. The inputs to this system are the distance measurements d_k, and these distance measurement will be used to update the state of the object as given in step 4 of the Kalman filter procedure in Table 1.
The process noise covariance matrix Q accounts for the unmodeled factors of the system that will be treated as random noise. For example, in the systems of equations above, while the change of velocity is accounted for by acceleration, the change in acceleration is not considered. The contribution of this effect to the state is accounted for as random noise. See (Welch, et al., 2007) for a more in depth discussion. In this example, q_x and $q_y$can be considered as the standard deviations of the acceleration noise in the x and y direction, respectively.

$$Q = \begin{bmatrix} q_x\frac{\partial t^5}{20} & 0 & q_x\frac{\partial t^4}{8} & 0 & q_x\frac{\partial t^3}{6} & 0 \\ 0 & q_y\frac{\partial t^5}{20} & 0 & q_y\frac{\partial t^4}{8} & 0 & q_y\frac{\partial t^3}{6} \\ q_x\frac{\partial t^4}{8} & 0 & q_x\frac{\partial t^3}{3} & 0 & q_x\frac{\partial t^2}{2} & 0 \\ 0 & q_y\frac{\partial t^4}{8} & 0 & q_y\frac{\partial t^3}{3} & 0 & q_y\frac{\partial t^2}{2} \\ q_x\frac{\partial t^3}{3} & 0 & q_x\frac{\partial t^2}{2} & 0 & q_x\partial t & 0 \\ 0 & q_y\frac{\partial t^3}{3} & 0 & q_y\frac{\partial t^2}{2} & 0 & q_y\partial t \end{bmatrix} \quad (4)$$

At time step k, the three measured distances d_{ik} (i=1, 2, 3) to the locations of the three beacons (x_i, y_i), can be used to relate the location of the target object (x_k, y_k) using the following equation:

$$\begin{aligned} d_{1k} &= \sqrt{(x_k - x_1)^2 + (y_k - y_1)^2} + \tilde{d}_{1k} \\ d_{2k} &= \sqrt{(x_k - x_2)^2 + (y_k - y_2)^2} + \tilde{d}_{2k} \\ d_{3k} &= \sqrt{(x_k - x_3)^2 + (y_k - y_3)^2} + \tilde{d}_{3k} \end{aligned} \quad (5)$$

where $\tilde{d}_{1k}$, $\tilde{d}_{2k}$ and $\tilde{d}_{3k}$ are distance measurement errors. The equation (5) can be expressed as (6).

$$\begin{bmatrix} d_{1k} \\ d_{2k} \\ d_{3k} \end{bmatrix} = H \cdot \begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \\ \ddot{x}_k \\ \ddot{y}_k \end{bmatrix} + \begin{bmatrix} \tilde{d}_{1k} \\ \tilde{d}_{2k} \\ \tilde{d}_{3k} \end{bmatrix} \quad (6)$$

Where H is the measurement matrix that relates the current state to the output. Since the output equations (5) are non-linear, the Jacobian needs to be used, where

$$H = \begin{bmatrix} \frac{\partial d_1}{\partial x} & \frac{\partial d_1}{\partial y} & 0 & 0 & 0 & 0 \\ \frac{\partial d_2}{\partial x} & \frac{\partial d_2}{\partial y} & 0 & 0 & 0 & 0 \\ \frac{\partial d_3}{\partial x} & \frac{\partial d_3}{\partial y} & 0 & 0 & 0 & 0 \end{bmatrix} \quad (7)$$

and the partial derivatives are given in (8) and (9) below.

$$\frac{\partial d_i}{\partial x} = \frac{x - x_i}{\sqrt{(x - x_i)^2 + (y - y_i)^2}} \quad (8)$$

$$\frac{\partial d_i}{\partial y} = \frac{y - y_i}{\sqrt{(x - x_i)^2 + (y - y_i)^2}} \quad (9)$$

The assumption that, the measurement noise associated with the distance measurements of a beacon is independent among the three beacons, will be made. This will result in measurement noise values for the appropriate beacon only along the diagonal of the measurment noise matrix R.

$$R = \begin{bmatrix} MN_1 & 0 & 0 \\ 0 & MN_2 & 0 \\ 0 & 0 & MN_3 \end{bmatrix} \quad (10)$$

Where MN_1, MN_2, and MN_3 are the measurement noises to beacon 1, 2 and 3, respectively. Using the formulation of the problem as described above, the following equations can be evaluated iteratively to track the target object. In each iteration, five steps are performed, as shown in Table 1. The current state X_{k-1} is used to estimate the location at the next time instant. The error covariance matrix P_k^- in the next time step is also projected using the state space model A and the process noise matrix Q in step 2. In step 3, the Kalman gain K_k is computed. The Kalman gain is used in step 4, when the distance measurements $D_k = [d_{k1}, d_{k2}, d_{k3}]^T$ from the beacons to the target object are obtained and are used to update the state X_k. The current position (x_k, y_k) is a subset of the state vector X_k.

Procedure of Extended Kalman Filter		
1. Project the state ahead	$X_k^- = A \cdot X_{k-1} + B \cdot U_k$	(11)
2. Project the error covariance ahead	$P_k^- = A \cdot P_{k-1} \cdot A^T + Q$	(12)
3. Compute the Kalman gain	$K_k = P_k^- \cdot H^T \cdot (H \cdot P_k^- \cdot H^T + R)^{-1}$	(13)
4. Update estimation with measurements	$X_k = X_k^- + K_k \cdot (D_k - H \cdot X_k^-)$	(14)
5. Update the error covariance	$P_k = (I - K_k \cdot H) \cdot P_k^-$	(15)

Table 1. Procedure of Extended Kalman Filter

Figure 3 depicts the iterative process described in Table 1. The current state and the error covariance matrix are projected for the next time step using equations (11) and (12). The next state is then computed by correcting the estimate that was made in step 1. This is done by using the measurement noise matrix R to compute the Kalman gain K. The Kalman gain K is used to scale the contribution of the distance measurement inputs to the next state estimate in step 4. The error covariance matrix that was projected for the next time step is corrected, and the process repeats all over again.

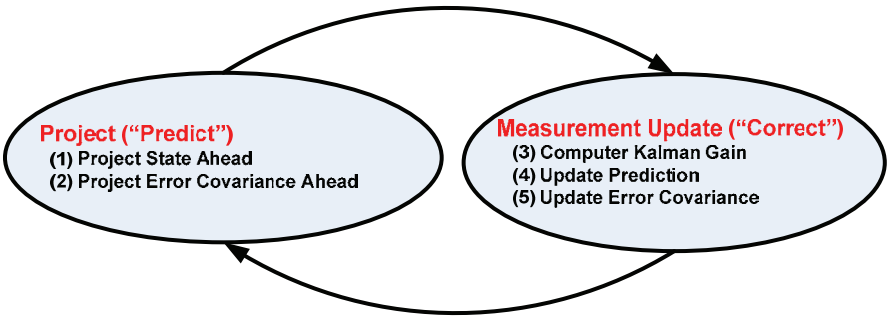


Fig. 3. Extended Kalman filter operation

In this chapter, the performance of the *P*, *PV*, and *PVA* extended Kalman filter models will be compared. The benefit of one over the other depends upon the characteristics of the motion of the object. A system modeled using just *P* will work when the position is mostly constant and the velocity can be treated as noise. In the case of a *PV* model, it will tend to work better when velocity is mostly constant, and the acceleration can be treated as noise. *PVA* on the other hand works better when the acceleration is mostly constant (Welch, et al., 2007).

4. Experiment design

We will explore the performance of the *P*, *PV*, and *PVA* models of the Kalman filters using MIT's Cricket sensors (Priyantha, et al., 2000). However, as was discussed, the use of ultrasound introduces noise. The distance measurements returned by the sensors fluctuate often and these measurements are the inputs to the Kalman filters. We simulate the two dimensional motion of an object by collecting distance measurements of the mobile node while physically moving it in a network composed of Cricket sensors (see Figure 4). A tile floor provided a very regular grid of 30 cm in size upon which the sensors could be accurately located. Four Cricket sensors are used, three as beacons and the other as the mobile node.

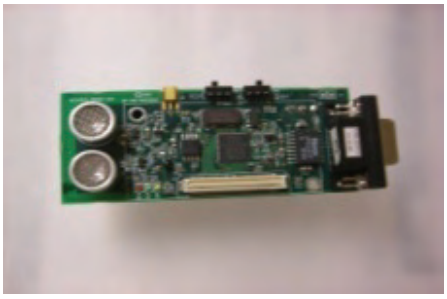


Fig. 4. Cricket sensor

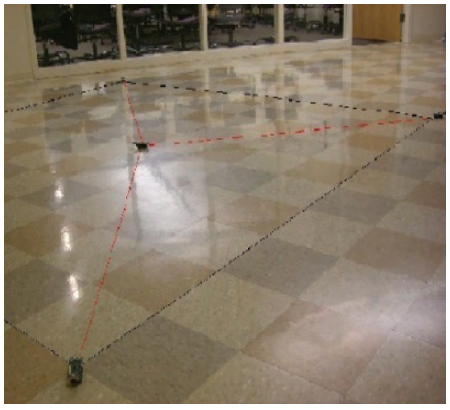


Fig. 5. Experiment Test Bed

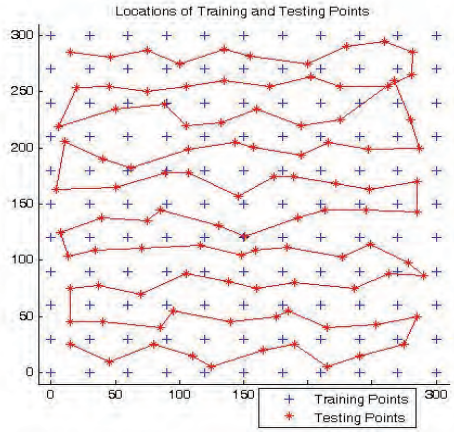


Fig. 6. Locations of Testing Data Collected

Using a grid of 300 cm × 300 cm, beacons were placed at the coordinates of (0, 300), (300, 300), and (300, 0) of the grid as shown in Figure 5. For each of the 121 tested position, distances measurements were collected from the three beacons. This is indicated using the "*" sign in Figure 6. The distances are input as D_k in step 4 of the Kalman filter procedure to update the state estimate, as shown in Table 1.

The measurement noise for each of the sensors was assumed to be independent of the others. The measurement noise was obtained by taking the average difference between the actual and estimated distances from the beacons to the nodes. However, it should be noted that for the path generated in Figure 6, the measurement noise was restricted to 30 cm. Any measurement with error greater than this was discarded. In the next section, the robustness of the Kalman filter will be analyzed using the raw distance measurements of another path traced out by a moving object.

The process noise matrix Q is more difficult to obtain. Approximate behavior such as the standard deviation of the position for the P model about the estimated movement of the object can be used. Of course, the standard deviation of velocity would be applied for the PV model, and the standard deviation of the acceleration for the PVA model.

Determination of the correct process noise parameters are key to accurate localization as will be seen in the next section. The process noise parameters for comparing the three models of the Kalman filter were obtained by computing the average error in localization for varying values of process noise, and the noise parameters that correspond to the least localization error were used.

5. Results

5.1 Performance comparison

Figures 7A, 7C, and 7E show the localization accuracy of the P, PV, and PVA methods, respectively. Two metrics were utilized in comparing the performances of the Kalman filters. The first is the average distance error in localization per estimate, as defined by equation (16).

Another metric that was used is the Root Mean Square Error (RMSE). This metric computes the error in localization for the X and Y coordinates and squares it. The sum of all errors are computed, divided by the number of estimates, and the square root is taken of the resulting value. The benefit of the RMSE given in equation (17) is that the error in localization of the X and Y coordinates is available. The X and Y RMSE values can be combined using equation (18) to result in the Net RMSE that describes the net error. An interesting characteristics of the RMSE is that it is biased towards large errors. A large error make a larger contribution in RMSE than in average distance error.

$$Distance\ Error = \frac{\sum\sqrt{(X_{Actual} - X_{Est})^2 + (Y_{Actual} - Y_{Est})^2}}{Number\ of\ Estimates} \tag{16}$$

$$RMSE = \sqrt{\frac{\sum(Actual - Estimated)^2}{Number\ of\ Estimates}} \tag{17}$$

$$Net\ RMSE = \sqrt{{X_{RMSE}}^2 + {Y_{RMSE}}^2} \tag{18}$$

As Table 2 indicates, the P model has the least distance error per estimate and the least Net RMSE and hence the best localization performance. This is followed closely by the PV

model, and finally the *PVA* model. It should be noted, that Table 2 lists the minimum error that can possibly be attained by these methods. This was done, by evaluating the Kalman filter for a variety of process noise parameters and selecting the resulting minimum error. The discussion that follows will clarify this point.

Method	Avg. Distance Error Per Estimate (cm)	RMSE (cm)	Net RMSE (cm)
P	8.1626	(6.8520, 7.2268)	9.9587
PV	8.5811	(6.8160, 7.5349)	10.1603
PVA	9.4023	(7.7226, 7.9404)	11.0741

Table 2. Accuracy Comparison between *P*, *PV* and *PVA* models

There is an interesting anomaly between the RMSE values of the X coordinates for the *P* and *PV* models. Although the *P* model has the lower distance error per estimate and Net RMSE, its localization in the X direction is not as good as the *PV* model. This anomaly indicates that the *PV* model can potentially be a better candidate in some scenarios.

A good portion of the error for all of the methods seems to be along the edge of the testing boundary in the vicinity of the beacons. This may be due to the fact that the use of ultrasound on the Cricket sensors results in large interferences between signals close to the beacons.

The experiments also show that the Kalman filters display relatively large errors at the edges of the boundaries. This may be due to the fact that the Kalman filters iteratively close in on the localized position. At the boundaries, where the object's motion takes a sudden turn, the Kalman filter's estimates require a few iterations before it can "catch up" with the object. This may be due to the process noise of the simulated motion of the object not adhering to Gaussian characteristics, which is assumed in Kalman filter.

In the case of the *P* model, the assumption that velocity and acceleration are just random noise allow it to take these turns with less error. The *PV* and *PVA* models are forced to maintain rigidly to the Kinematic equations.

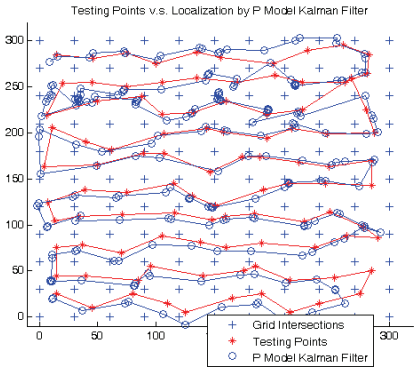
Figures 7B, 7D, and 7F depict the percentage of varying magnitudes of error for the *P*, *PV*, and *PVA* models respectively. It is observed that the *P* model has a greater percentage of errors of low magnitude, whereas the *PV* and *PVA* models have lower percentages, but of greater magnitude. The *P* model seems to make many small errors at each step, whereas the *PV* and *PVA* make large errors, typically at the edge of the boundary.

Table 3 summerizes the maximum error values and their locations in our experiments. Note that the *PV* and *PVA* models have their maximum error values close to the same locations.

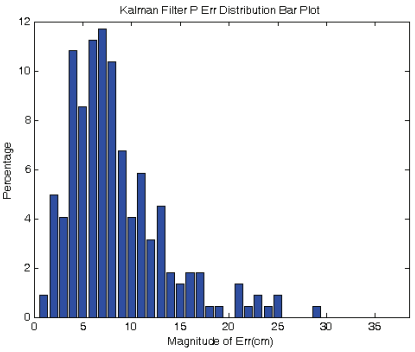
Method	Max Error (cm)	Coordinate of Max Error
P	38.0901	(285, 50)
PV	31.7478	(20, 254)
PVA	32.6776	(45, 255)

Table 3. Maximum error and location of maximum error

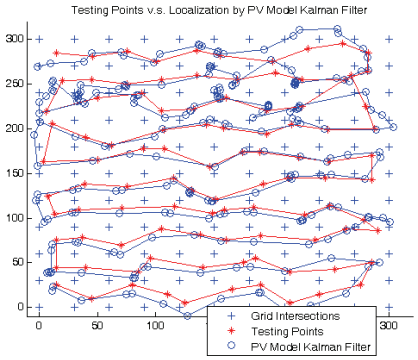
The process noise is an independent variable of the system, and it attempts to predict the motion and future estimates of the position of the object. For this reason, it is impossible to exactly ascertain this paramter. Even when points on the path of the object are given, this is not sufficient since they are just samples of a coarse simulation of the movement of an object.



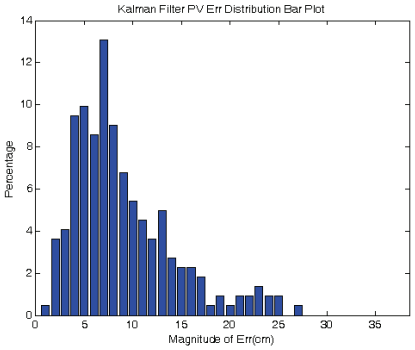
A. *P* model estimation



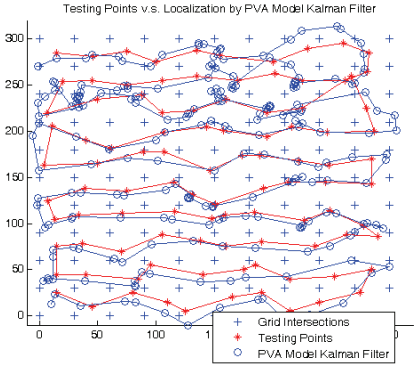
B. *P* model error distribution



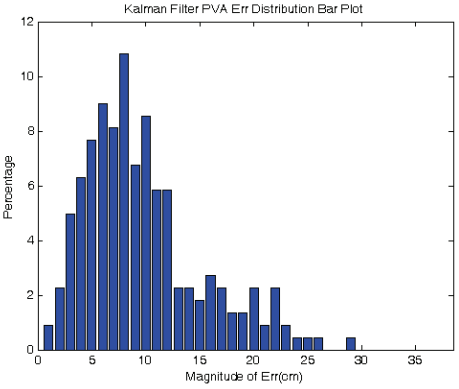
C. *PV* model estimation



D. *PV* model error distribution



E. *PVA* model estimation



F. *PVA* model error distribution

Fig. 7A, 7C, 7E. Comparison between actual and estimated tracked paths as given by *P*, *PV*, and *PVA* models

Fig. 7B, 7D, 7F. Distribution of the magnitude of error for localized positions for P, PV, and PVA models.

The process noise parameter matrix Q was obtained by using varying values of the X and Y process noise parameters, and computing the average distance error for the entire path for that set. Figure 8 depicts a surface plot of X and Y process noise parameters versus the resulting average distance error. It is interesting how the use of inaccurate process noise parameters effects the perfomance; for low value of the X component of the process noise, the resulting error is very high. The minimum average distance error as given in Table 2, was found for process noise paramters, (300,100), which interestingly implies that the process noise parameters don't seem to matter a great deal for the P model. In this model, the velocity is considered as the process noise, and hence the units can be considered as cm. The minimum error that was listed in Table 2 was obtained for the process noise parameters listed in Table 4.

The same steps were taken to obtain the process noise parameters for the PV model. However, here the acceleration is considered as the process noise, and the unit are cm/sec.

Method	Avg Distance Error Per Estimate (cm)	Process Noise	Units
P	8.1626	(300,100)	cm
PV	8.5811	(300,62)	cm/sec
PVA	9.4023	(37,18)	cm/sec^2

Table 4. Process noise parameters corresponding to the minimum distance error

The PVA model presents an interesting relationship between the process noise and its performance unlike the P and PV models. The process noise parameters for the PVA model corroborate well with the movement of the object, since the object moves almost 30 cm along the X-axis and almost 20 cm along the Y-axis at each step as depicted in Figures 7A, 7C, and 7E.

The process noise values listed in Table 4 are input to the Kalman filter by setting q_x of the process noise matrix Q in equation 4 to the x component, and q_y to the y component.

In the analysis presented above, the average distance error in localization was used as a metric in determining the process noise paramters. However, it is also possible to use the RMSE value to do the same. Suprisingly, there is a strong correlation between the process noise obtained using the average distance error and RMSE metrics indicating the suitability of this procedure. Table 5 lists the appropriate process noise values corresponding to the minimum RMSE value. In the interest of space and brevity, the surface graphs that were used to obtain these values will not be displayed.

Method	RMSE (cm)	Process Noise	Units
P	9.9587	(300,100)	cm
PV	10.1572	(300,71)	cm/sec
PVA	11.0584	(34,13)	cm/sec^2

Table 5. Process noise parameters corresponding to the minimum RMSE

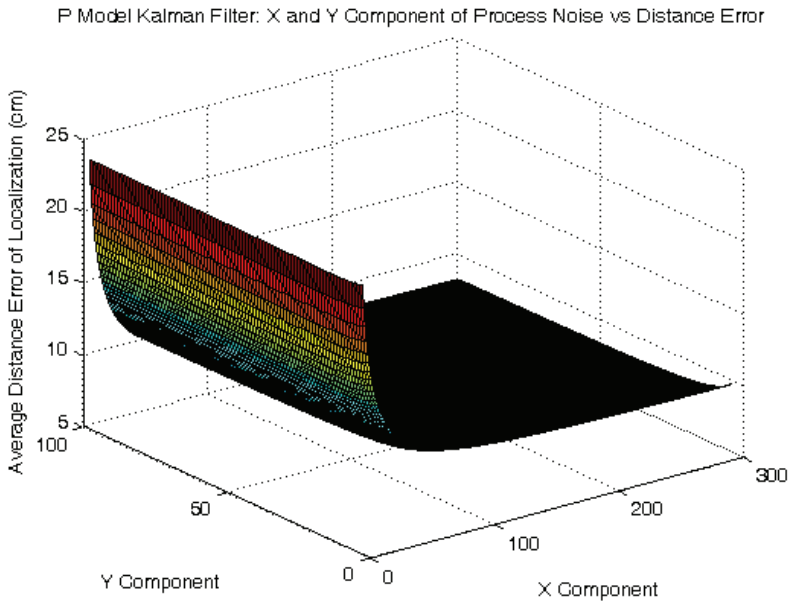


Fig. 8. Process noise versus average distance error of localization for *P* model.

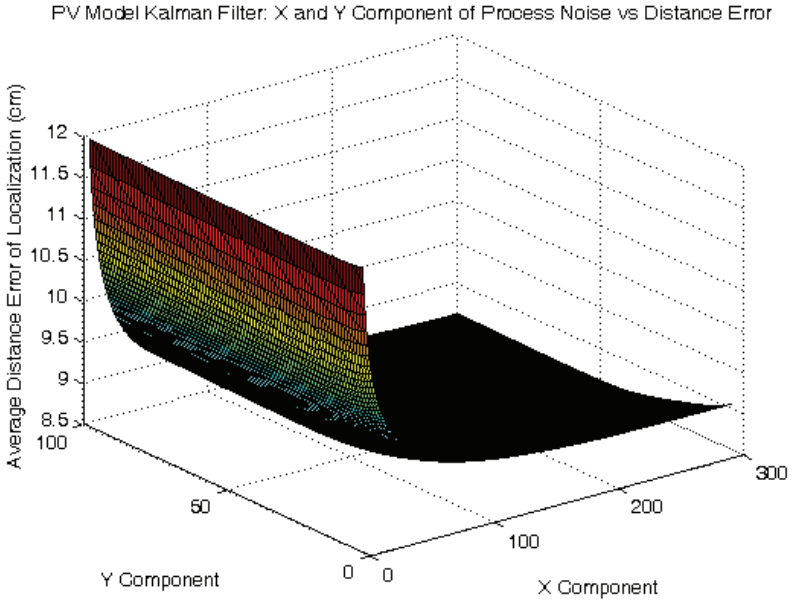


Fig. 9. Process noise versus average distance error of localization for *PV* model

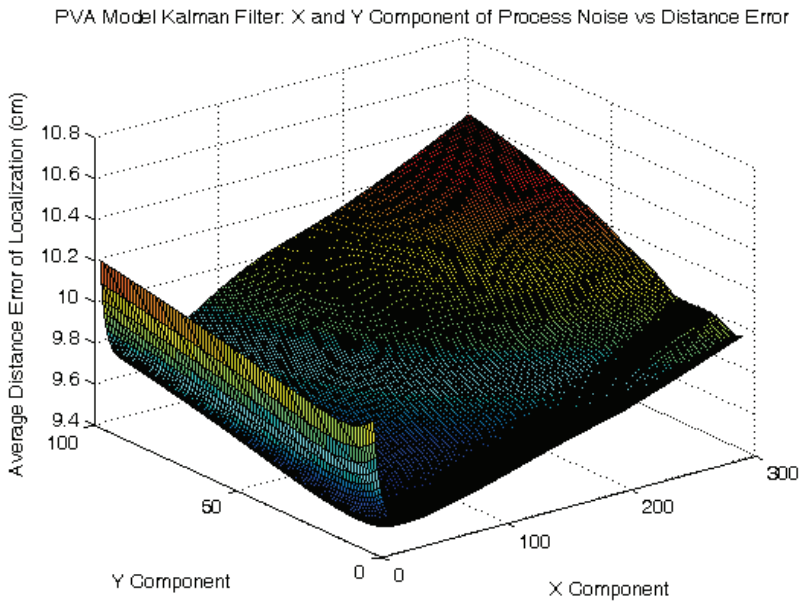


Fig. 10. Process noise versus average distance error of localization for *PVA* model

5.2 Robustness comparison

Armed with the analysis procedure given above , the robustness of these methods will now be evaluated. A new path traced out by a moving node was generated as shown in Figure 11 that contains nearly 271 points with an average step size of 10.48 cm whereas the path in Figure 6 contains 222 points with an average step size of 14.13 cm.

The robustness of each of these Kalman filter models was examined by comparing the performance of each of these models for various magnitudes of erroroneous distance measurements. This is unlike the first path generated, where the error in distance measurements exceeding 30 cm was discarded. The distance measurements to the path traced out by the moving object as shown in Figure 11 were collected and categorized into six data sets. In the first data set, no distance measurements were removed. All spurious distance measurements of any magnitude were kept. The maximum threshold for the error was set to 10,000 cm to include all distance measurements regardless of the size of the error. (No distance measure error exceeded 10,000 cm.) In the second set of data, distance measurements with errors of only 300 cm or less were kept. In the third, distance measurements with errors of only 200 cm or less were kept. In the fourth, distance measurements with errors of 100 cm or less were kept. In the fifth, distance measurements with errors of 50 cm or less were kept, and in the last data set, only measurements with errors that did not exceed 30 cm were kept. In this way, the method most capable of localizing under these demanding conditions can be identified.

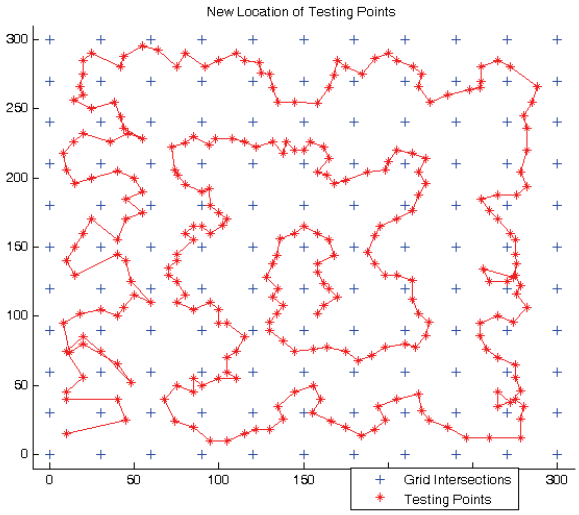


Fig. 11. New path generated for testing robustness

Error Method	Maximum Error Threshold (cm)	*P* Model		*PV* Model		*PVA* Model	
		Avg Error (cm)	Process Noise (cm)	Avg Error (cm)	Process Noise (cm/s)	Avg Error (cm)	Process Noise (cm/s²)
Dist. Error	10000	278.6226	(6,11)	328.0187	(8,7)	479.274	(5,5)
RMSE	10000	376.3137	(6,5)	480.9813	(7,6)	645.7573	(5,6)
Dist. Error	300	22.149	(16,26)	28.039	(72,50)	35.0343	(6,38)
RMSE	300	29.6411	(5,10)	46.7252	(34,5)	59.0939	(6,26)
Dist. Error	200	15.4266	(35,100)	17.5227	(10,5)	19.2767	(8,9)
RMSE	200	20.5447	(8,20)	24.5717	(5,5)	27.8166	(8,9)
Dist. Error	100	11.8554	(69,100)	12.6885	(150,67)	13.3489	(38,5)
RMSE	100	14.5468	(33,100)	15.7661	(10,71)	16.2669	(8,5)
Dist. Error	50	10.9748	(82,100)	11.4858	(300,100)	11.9595	(107,52)
RMSE	50	12.7345	(63,100)	13.6644	(300,100)	14.2215	(121,5)
Dist. Error	30	10.5008	(125,100)	10.9549	(300,100)	11.2604	(174,50)
RMSE	30	11.8822	(100,100)	12.6447	(300,100)	13.0828	(300,10)

Table 6. Six different data sets and the minimum averge distance error and the RMSE error and the corresponding process noise parameters for the *P*, *PV*, and *PVA* models.

Figure 12, 13, and 14 depict the surface plots of the process noise versus the average distance error measurement of localization for the *P*, *PV*, and *PVA* models. In Figure 12, the average distance error increases at a nearly smooth and steady rate. This is in contrast to Figure 8, where the error decreased for increasing process noise. In Figure 13 and 14, the average distance error flutuates wildly with changing process noise, while the general trend seems to indicate that the error increases with increasing process noise.

Surface plots examining the behaviour of the RMSE instead of the average distance error were also generated, however, for the sake of brevity they will not be presented here. The first rows in Table 6 lists the minimum average distance error, 278.6226 cm, for process noise at (6,11) cm for the set of data containing up to 10,000 cm error in distance measurements. The minimum average distance errors and the corresponding process noise values for the *PV* and *PVA* models are listed next. The next row in Table 6 lists the minimum RMSE values and corresponding process noise parameters for the *P*, *PV*, and *PVA* models respectively. The next two rows deal with the minimum average distance error and RMSE for a maximum of 300 cm error in the distance measurements for the *P*, *PV*, and *PVA*. The results for maximum thresholds of 200, 100, 50, and 30 cm follow.

It is clear from the graph that the *P* model consistently has lower error than the other models, and the *PV* model performs better than the *PVA*. It is also interesting to note that at a high error threshold, the process noise for the *P*, *PV* and *PVA* models is low. As the error threshold decreases, the process noise increases.

It is a testament to the Kalman filters that the error rates between error threshold 100 cm and 30 cm for the *P*, *PV*, and *PVA* does not change much. For example, the average distance error for *P* changes only 1.3546 cm (11.8554 – 10.5008) and the error rates for *PV* and *PVA* change by only 1.7336 and 2.0885 cm. That is at most a 16 percent drop in error for nearly a 70 percent decrease in the magnitude of erroneous distance measurements. In other words, even with errors of up to 100 cm in distance measurments, the Kalman filter models were still capable of localizing fairly accurately. Figures 15A, 15C, and 15E reveal the performance of the *P*, *PV*, and *PVA* models for error threshold of 100 cm. Figures 15B, 15D, and 15F reveal the performance of the *P*, *PV*, and *PVA* models for error threshold of 30 cm. It is apparent between Figures 15A and 15C that the *P* model is better adept at tracking the irratic motion of the object. The *PV* and *PVA* models overshoot sudden turns and require several iterations before they close in on the object's position again.

Table 6 show that pre-filtering of the distance measurements can result in a significant decrease of errors in localization, especially between error thresholds of 10,000 and 100 cm. The level of pre-filtering does not need to be very rigorous since the Kalman filter is very robust. Extremely large errors in distance measurements can be easily spotted and eliminated. One such method is utilizing the past distance measurements and the maximum rate at which the object is expected to move. This projection can be used as a threshold against which the current distance measurements can be compared and if necessary discarded. Assuming that the last distance measaments for three becons are stored in d_1, d_2, and d_3. Maximum velocity and acceleration attainable by the object are v_{max} and a_{max} . Using the distance from beacon 1, the mobile node can either be moving away from the beacon or approaching it. In case that the object is moving away from the beacon, the distance from the beacon will increase as given by equation (19) for $d_{1_{max}}^{+}$ where T is the time step. However, if the object is approaching the beacon, then the distance will decrease as given by equation (20) for $d_{1_{max}}^{-}$.

$$d_{1_{max}}^{+} = d_1 + T \cdot v_{max} + \frac{1}{2} T^2 \cdot a_{max} \tag{19}$$

$$d_{1_{max}}^{-} = d_1 - T \cdot v_{max} + \frac{1}{2} T^2 \cdot a_{max} \tag{20}$$

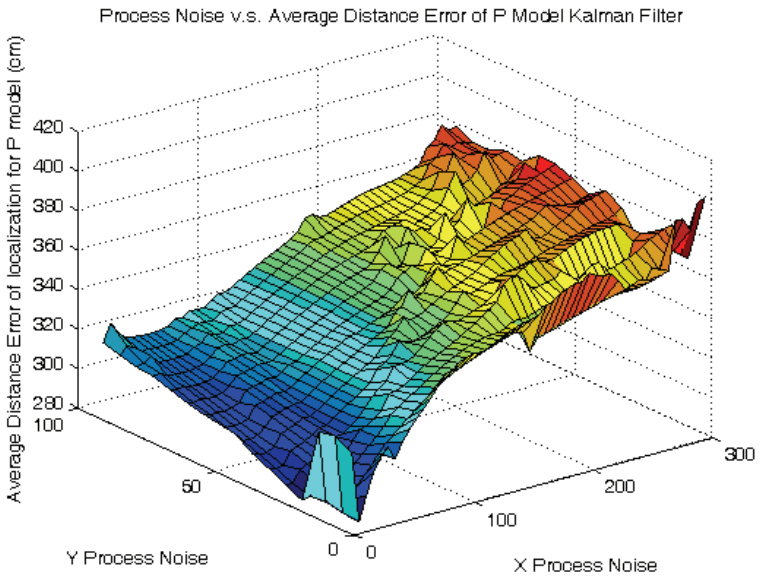


Fig. 12. Process noise versus localization error for *P* model using a maximum error threshold of 10,000 cm

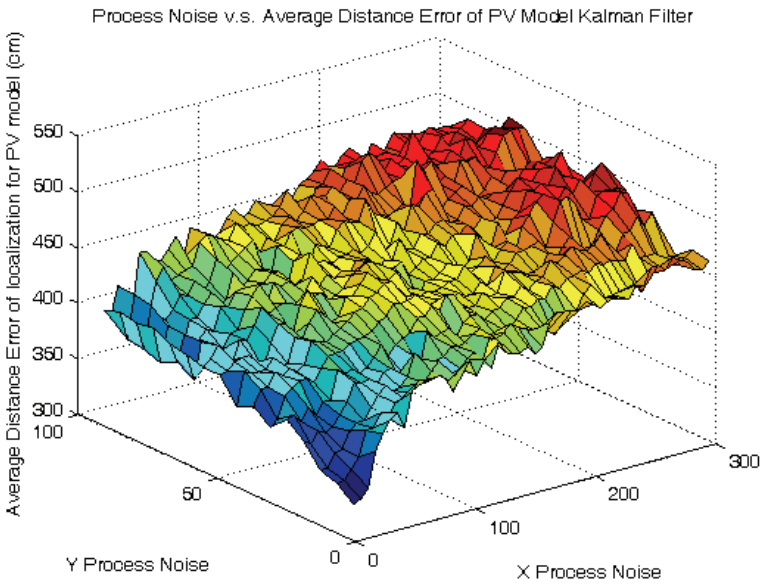


Fig. 13. Process noise versus localization error for *PV* model using a maximum error threshold of 10,000 cm

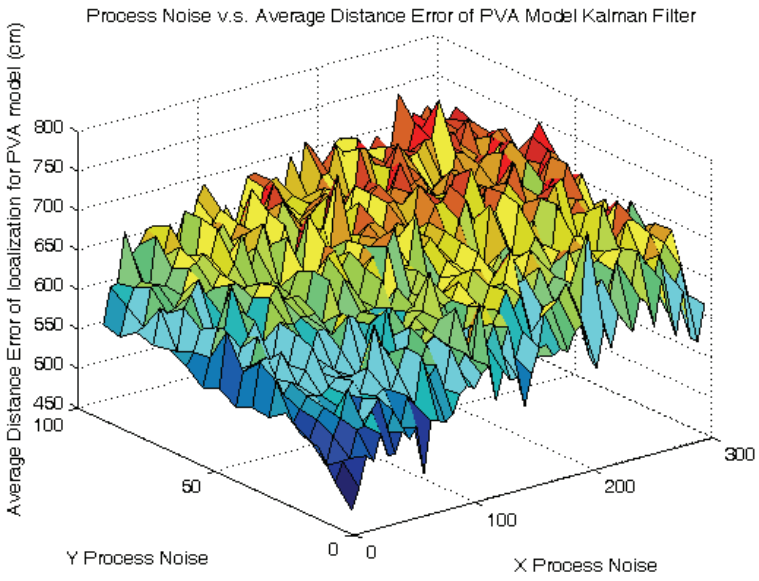


Fig. 14. Process noise versus localization error for PVA model using a maximum error threshold of 10,000 cm

Subtracting equation (20) from (19) results in:

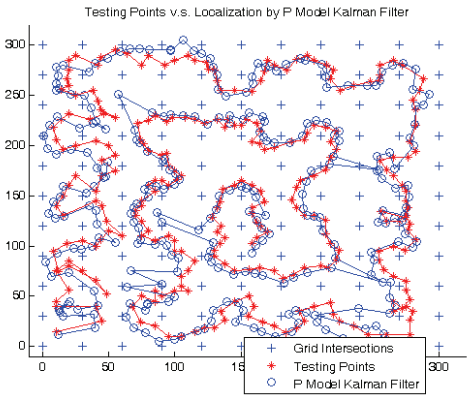
$$\begin{aligned}\Delta d_1 &= {d_1}^{+}_{max} - {d_1}^{-}_{max} \\ &= \left(d_1 + T \cdot v_{max} + \tfrac{1}{2}T^2 \cdot a_{max}\right) - \left(d_1 - T \cdot v_{max} + \tfrac{1}{2}T^2 \cdot a_{max}\right) \\ &= 2 \cdot T \cdot v_{max} + T^2 \cdot a_{max}\end{aligned} \tag{21}$$

Equation (21) results in a threshold against which we can compare future distance measurements from beacon $B1$. Thresholds for the other beacons 2 and 3 can be obtained similarly. If the absolute value of the current distance measurement minus the previous measurement is greater than than the predefined threshold ΔD, then this current distance measurement can be discarded.

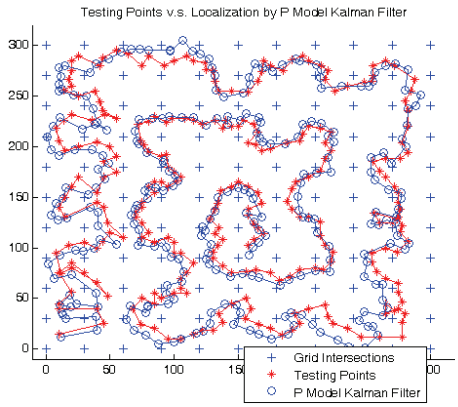
5.3 Computation requirement comparison

Thus far, only the accuracy of the localization methods has been examined without any discussion of the computation requirements associated with them. As mentioned before, these localization methods will be implemented on an embedded system with limited capabilities. Based on the application, an appropriate localization method must be used that balances accuracy with the capabilities of the system.

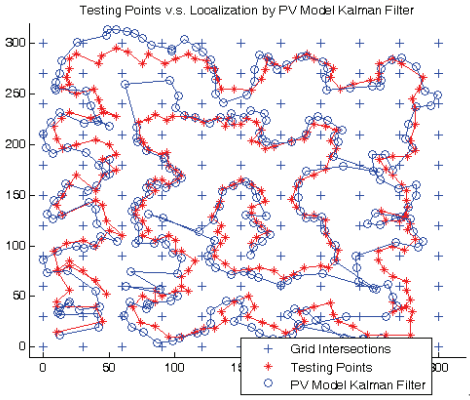
The following analysis utilizes the number of floating point operations as a metric to compare the different methods. For simplicity of presentation, this analysis assumes that the embedded system has native floating point capabilities and does not rely on integer computations to mimic floating point operations. Further, this analysis only accounts for



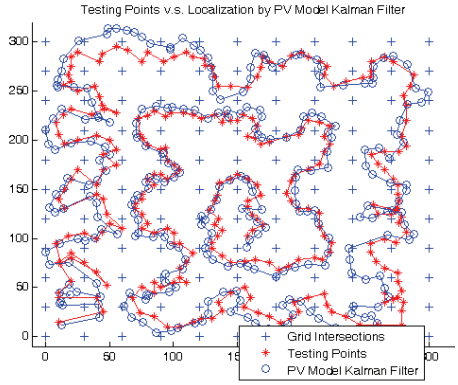
A. *P* model, Error threshold = 100 cm



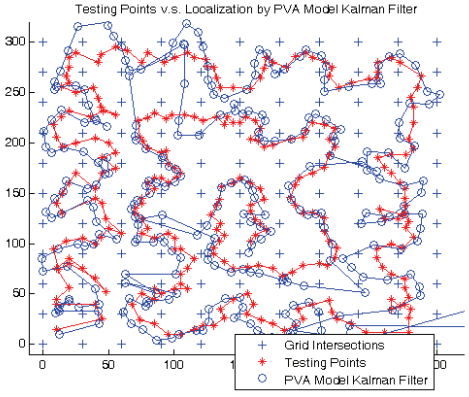
B. *P* model, Error threshold = 30 cm



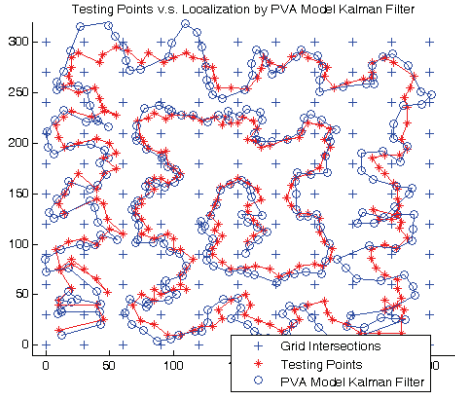
C. *PV* model, Error threshold = 100 cm



D. *PV* model, Error threshold = 30 cm



E. *PVA* model, Error threshold = 100 cm



F. F. *PVA* model Error threshold = 30 cm

Fig. 15A, C, and E. Performance of *P*, *PV*, and *PVA* models with error threshold of 100 cm.

Fig. 15B, 15D, and 15F. Performance of *P*, *PV*, and *PVA* models with error threshold of 30 cm.

steady-state computation, meaning the initialization and setup computations are not considered. All the addition, subtraction, multiplication, and division steps that must be evaluated for each matrix operation for the equations given in Table 1 were considered.

Method	Number of Floating Point Operations per Iteration
P	268
PV	884
PVA	2220

Table 7. Comparison of Floating Point Operations between methods

As Table 7 reveals, the P model is the least computationally intensive. It is followed by the PV and PVA Kalman filters. This is intuitive since the P model has only two element in its state, and the dimensions of the state space model A, process noise Q, covariance P_k, and Kalman gain $K_k$matrices will be much smaller. Whereas the larger number of elements in the PV and PVA state result in greater dimensional matrices used during the evaulation of the Kalman filter equations, and hence a greater number of computations.
The Kalman filter equations involve many matrix multiplications and an inverse operation for computing the Kalman gain K. These two operations have complexity O(n^3) and as a result the Kalman filter is also of complexity O(n^3) where n is the number of parameters in the state.

Method	Order of Magnitude	Comment
P	O(n^3)	n is the number of elements in the state variable.
PV	O(n^3)	n is the number of elements in the state variable.
PVA	O(n^3)	n is the number of elements in the state variable.

Table 8. Computational complexity between methods

5.4 Memory requirement comparison

This section takes a cursory glance at the memory needs of the three Kalman filter models. It should be noted that the memory usage described here is the steady state memory, this does not take into account any initializations that may be required for different applications. It is also assumed that floats are four bytes long and that all variables in the Kalman filter equations require the float type.
In the expressions for the three Kalman filters, n is the number of elements in the state variable and m is the number of distance readings. The expressions for the memory requirements of the Kalman filters include an additional ($n^2 + n + 2nm + m + m^2$) bytes of memory for temporary variables.
Since the measurement noises for each of the sensors in the experiment was assumed to be independent, R (measurement noise matrix) in Equation (12) of the Kalman filter contains values only along the diagonal. Instead of a matrix, a vector (mx1) can be used to represent each of the values along the diagonal and saving some space.

Name of Variable	Kalman Filter Variable	Matrix Dimension	Total Size
State model	A	$n \times n$	n^2
State estimate	X_k^-	$n \times 1$	n
Last state estimate	X_{k-1}	$n \times 1$	n
Estimated covariance	P_k^-	$n \times n$	n^2
Last estimated covariance	P_{k-1}	$n \times n$	n^2
Process noise	Q	$n \times n$	n^2
Output measurement matrix	H	$n \times m$	nm
Measurement noise	R	$m \times 1$	m
Kalman gain	K_k	$n \times m$	nm
Next state	X_k	$n \times 1$	n
Distance vector	D_k	$m \times 1$	m
Next covariance	P_k	$n \times n$	n^2
Position of sensors	Sensor location	$m \times 2$	$2m$
		Total	$5n^2 + 3n + 2nm + 4m$

Table 9. Computation of memory usage of Kalman filter variables.

The temporary variables listed in Table 10 are used during the Kalman filter operation to hold intermediate values during matrix multiplication and other operations. It should be noted that during the computation of the error covariance matrix P_k, it requires a temporary variable of size($n \times n$). However, while projecting the error covariance matrix P_k^-, a temporary variable of size ($n \times n$) is utilized. This same temporary variable can be used during the computation of P_k as well.

$$\begin{aligned} \text{Total Memory} &= [\text{Kalman Filter Variables} + \text{Temporary Variables}] \times 4 \text{ Bytes} \\ &= [(5n^2 + 3n + 2nm + 4m) + (n^2 + n + 2nm + m + m^2)] \times 4 \text{ Bytes} \\ &= [6n^2 + 4n + 4nm + 5m + m^2] \times 4 \text{ Bytes} \end{aligned} \quad (22)$$

Equation (22), describes the total memory required for the computation of the Kalman filter equations. The number of distance measurements m is assumed to be 3, since this is the smallest number of distance measurements required to localize. The value for n for each of the *P*, *PV*, and *PVA* models depends upon the number of elements in each of their states which is outlined in Table 11.

Kalman Filter Equations		Terms Requiring Temporary Variables	Variable Size	Total Size
$X_k^- = A \cdot X_{k-1} + B \cdot U_k$	(10)			
$P_k^- = A \cdot P_{k-1} \cdot A^T + Q$	(11)	$\{A \cdot P_{k-1}\}$	$n \times n$	n^2
$K_k = P_k^- \cdot H^T \cdot (H \cdot P_k^- \cdot H^T + R)^{-1}$	(12)	$\{P_k^- \cdot H^T\}$ $\{H \cdot P_k^-\}$ $\{H \cdot P_k^- \cdot H^T\}$	$n \times m$ $m \times n$ $m \times m$	nm nm m^2
$X_k = X_k^- + K_k \cdot (D_k - H \cdot X_k^-)$	(13)	$\{H \cdot X_k^-\}$ $\{K_k \cdot (D_k - H \cdot X_k^-)\}$	$m \times 1$ $n \times 1$	m n
$P_k = (I - K_k \cdot H) \cdot P_k^-$	(14)	$\{K_k \cdot H\}$	$n \times n$	
		Total Size	$n^2 + n + 2nm + m + m^2$	

Table 10. Computation of memory usage of temporary variables for Kalman filter.

Method	Number of elements (n)
P	2
PV	4
PVA	6

Table 11. Number of elements in the state variable for each of the methods

Method	Total Memory Usage	Number of Bytes
P	$6n^2 + 4n + 4nm + 5m + m^2$	320
PV	$6n^2 + 4n + 4nm + 5m + m^2$	736
PVA	$6n^2 + 4n + 4nm + 5m + m^2$	1344

Table 12. Comparison between memory requirments between localization methods

6. Summary

The Kalman filter is very capable of localizing using noisy distance measurements in a wireless sensor network. The P model of the Kalman filter was found to have the best performance in these examples, however, depending on the motion of the tracked object the

PV or *PVA* model could be better. In the examples studied in this chapter, the *P* model is more adept at tracking the movement of an object with sporadic motion with sharp turns. Since the *P* model treats velocity and acceleration as random noise, it is less bound by these parameters. The *PV* and *PVA* models are bound by the Kinematics equations which do not allow for motion that is discontinuos in velocity and acceleration.

Two metrics were introduced that can be used to measure the accuracy of localiztion: the average distance error in localization and the Root Mean Square Error (RMSE). The average distance error in localization and the RMSE are very consistent metrics, although the RMSE method will return a higher error rate due to the squaring of the error.

The process noise parameters allow the Kalman filter to project the position of the object in the next time instant. If the proper process noise parameters are not used, the performance of the Kalman filters will be severly affected. In oder to obtain process noise parameters, the error rates can be computed for a variety of process noise paramters and the process noise parameters that result in minimum localization error can be used. This needs to be done each time a new path is expected to be taken by a mobile node because by comparing Table 4 and the section of Table 6 with error threshold of 30 cm, we can see that the process noise that results in the minimum localization error does indeed change between the two paths.

The Kalman filters are robust, and as the results indicated, there was very little difference in performance when the magnitude of errors in distance measurements was 100 cm and when the magnitude of error was 30 cm. However, despite the Kalman filters robustness, significant performance gains can be attained if the distance measurements used for localization are pre-filtered to ensure that they are not too erroneous. A method that was suggested was using the past distance measurements and the maximum velocity and acceleration attainable by the mobile node to project the next distance measurement. If the next distance measurement exceeds this threshold, it can be discarded.

Since localization methods well suited for wireless sensor networks using embedded systems with limited resources was desired, there is a need to analyze the computation and memory requirements. The *P* model Kalman filter was found to have the best computation and memory requirements due to the small number of elements in its state variable. This is followed by the *PV* model, and then the *PVA* model.

Although, in the experiments outlined in this chapter the process noise corresponding to reasonable low localization error was obtained, the task of obtaining process noise values in a real world scenario with a real moving object may be challenging. Especially if the path of the object is not known beforehand.

An assumtion that was made during these experiments was that the measurement noise *R* of each beacon is independent of the other which resulted in values only along the diagonal. However, in actuality, this is not the case especially when dealing with ultrasound pulses; the ultrasound pulses from each of the beacons may interfere with each other. This means that the measurement noise matrix, *R*, has values in all positions. Future work is needed to study the effects of interferances between beacons.

The experiments in this chapter focused only on the use of three beacons. Further study is needed to determine the effects of having more than three beacons and determine a policy for selecting the beacons to use the distance measurements from.

7. Acknowledgements

This research work is sponsored by National Science Foundation (NSF) IGERT-0504494, GK-12 0538457, CCF-0621493, and CNS-0723093.

8. References

Alippi, C.; Mottarella, A. & Vanini, G. (2005). A RF map-based localization algorithm for indoor environments, *Proceedings of IEEE International Symposium on Circuits and Systems*, 2005. ISCAS 2005, pp.652-655, ISBN: 0-7803-8834-8, Kobe, May 2005, IEEE CNF, Piscataway

Harter, A.; Steggles, P.; Ward, A. & Webster, P. (1999). The Anatomy of a Context-Aware Application , *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 59-68, ISBN:1-58113-142-9 , Seattle, August 1999, ACM Press, New York.

Moore, D.; Leonard, J.; Rus, D. & Teller, S. (2004). Robust distributed network localization with noisy range measurements, *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 50-61, ISBN:1-58113-879-2, Baltimore, November 2004, ACM Press, New York

Patwari, N. & Hero, A. (2003). *Using proximity and quantized RSS for sensor localization in wireless networks*, Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications, pp. 20-29, ISBN:1-58113-764-8, San Diego, September 2003, ACM Press, New York

Priyantha, B.; Chakraborty, A. & Balakrishnan, H. (2000). The Cricket location-support system, *Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 32-43, ISBN:1-58113-197-6, Boston, August 2000, ACM Press, New York

Shareef, A.; Zhu, Y.; & Musavi, M. (2008). Localization Using Neural Networks in Wireless Sensor Networks, *Proceedings of ACM First International Conference on Mobile Wireless Middleware, Operating Systems, and Applications*, pp. 1-7, ISBN:978-1-59593-984-5, Innsbruck, February 2008, ICST, Brussels

Smith, A.; Balakrishnan, H.; Goraczko, M. & Priyantha, N. (2004). Tracking moving devices with the cricket location system, *Proceedings of the 2nd international conference on Mobile systems, Applications, and Services*, pp. 190-202, ISBN:1-58113-793-1, Boston, June 2004, ACM Press, New York

Want, R.; Hopper, A.; Falcao, V. & Gibbons, J. (1992). The Active Badge Location System. *ACM Transactions on Information Systems*, Vol. 10, Issue 1., (January 1992), page numbers (91-102), ISSN:1046-8188

Ward, A.; Jones, A. & Hopper, A. (1997). A new location technique for the active office. *IEEE Personal Communications*, Vol. 4, Issue 5., (October 1997) page numbers (42-47), ISSN: 1070-9916

Welch, G.; Bishop, G.; Vicci, L.; Brumback, S.; Keller, K. & Colucci, D. (2001) High-Performance Wide-Area Optical Tracking: The HiBall Tracking System. *Presence: Teleoperators and Virtual Environments*, Vol. 10, Issue 1., (February 2001) page numbers (1-22), ISSN 1054-7460

Welch, G.; Allen, B.; Ilie, A. & Bishop, G. (2007). Measurement Sample Time Optimization for Human Motion Tracking/Capture Systems, *Proceedings of Trends and Issues in Tracking for Virtual Environments, Workshop at the IEEE Virtual Reality 2007 Conference*, ISBN: 978-3-8322-5967-9, Charlotte, March 2007, Shaker Verlag, Aachen